情報処理学会論文誌　データベース　Vol.6　No.3　13–28 (June 2013)

推薦論文

ブートストラッピングによるWebからの属性名抽出

野村 慎太郎[1,a)]　中根 史敬[1]　土方 嘉徳[1,b)]　西田 正吾[1,c)]

受付日 2012年12月20日, 採録日 2013年4月5日

概要：Web には大量の半構造化文書が存在するが，それらに書かれた情報はデータベースのように整形されていない．そこで，それらの情報を属性と属性値の形で抽出する研究（情報抽出）がさかんに行われてきた．これまで属性値の抽出については自動で獲得する研究が行われてきたが，属性名抽出の自動化を試みた例は少ない．従来の属性名抽出の研究の中にはクエリログや表構造を用いる研究があるが，本研究ではブートストラッピングアルゴリズムを用いて，任意のユーザが利用できる属性名抽出手法を提案する．また，抽出に必要なパラメータと抽出精度との関係についても調査する．

キーワード：情報抽出，半構造化文書，ブートストラッピング，属性名

Attribute Extraction from the Web by Boot Strapping Algorithm

Shintaro Nomura[1,a)]　Humitaka Nakane[1]　Yoshinori Hijikata[1,b)]
Shogo Nishida[1,c)]

Received: December 20, 2012, Accepted: April 5, 2013

Abstract: A large number of semi-structured documents exist on the web. To utilize information written on them, attribute names and attribute values have to be extracted and stored in a database. Recently, there are a lot of researches using boot strapping algorithm to extract attribute values. Some studies extract attribute names using query logs and table structures. This study aims at extracting attribute names by using boot strapping algorithm. We also examine a relationship between the parameter required for our method and its precision.

Keywords: information extraction, semi-structured document, boot strapping, attribute name

1. はじめに

Web 上には大量の半構造化文書（その多くは HTML 文書）がある．しかし，これらに記述された情報はデータベースのように整形されていないため，計算機はデータを正確に扱うことができない．たとえば，Web 上には多くのパソコンに関する情報が存在するが，ユーザが「メモリが 4 GB 以上で重量が 1.4 kg 未満の Windows 7 を搭載したパソコン」を検索しようとしても，計算機は「メモリ」,「重量」,「OS」に対応する値を記憶していないため，これらを満たす「パソコン」を返すことができない．

一方，リレーショナルデータベースにおいては，データの要素は属性名と属性値の組（上記の例では,「メモリ」という属性名に対して「4 GB」が属性値）で表現され，1 つのオブジェクトに関するデータの組はレコードと呼ばれる単位で登録される [1]．計算機が Web 文書に書かれた情報を理解するためには，それらを上記の形式で抽出する必要がある．具体的な課題としては，半構造化文書から，(1) 属性名を抽出すること，(2) 各属性に対する属性値を抽出すること，の 2 点がある．

これまで，前述 (2) の属性値抽出については，情報抽出の研究分野で多くの研究が行われてきた．具体的な手法

[1] 大阪大学大学院基礎工学研究科
Graduate School of Engineering Science, Osaka University, Toyonaka, Osaka 560–8531, Japan
[a)] nomura@nishilab.sys.es.osaka-u.ac.jp
[b)] hijikata@sys.es.osaka-u.ac.jp
[c)] nishiada@sys.es.osaka-u.ac.jp

本論文の内容は 2012 年 11 月の WebDB2012 にて発表され，同シンポジウムプログラム委員会により情報処理学会論文誌データベースへの掲載が推薦された論文である．

としては，(i) あらかじめ人手で作成した抽出ルール（またはテンプレート）を用いる手法 [2], [3] や，(ii) あらかじめ人手で文書中の抽出位置にタグ付けをしておき，機械学習により抽出ルールを学習する手法 [4], [5], [6]，(iii) 少数の抽出する値を人手で与えておき，抽出ルールと抽出値を交互に繰り返し学習していく手法（ブートストラッピング [7], [8], [9], [10]）がある．これらの手法を用いることにより，ニュース記事などの均質な文書だけでなく，Web 文書などの多様な形式の文書においても，ある程度の精度で属性値の抽出ができるようになってきている．

一方，前述（1）の属性名抽出については，属性値の抽出ほど多くの研究が行われているわけではない．多くの属性名抽出の研究は，フォームや表などを対象としている [11], [12], [13], [14], [15]．文献 [11], [12], [13] はあらかじめ人手で作成した抽出ルールを用いている．文献 [14], [15] は人手で表中のセルにラベル付けした事例から属性名を学習している．しかし，これらの手法は特定の記述形式を対象としており，Web から網羅的に属性名を獲得することができない．また，抽出ルールの作成や学習データの入力には人手を要する．そこで本研究では，人手を必要としないアルゴリズムであるブートストラッピングを用い，Web から網羅的にかつ自動で属性名を獲得することを目指す．

属性名は属性値と比較するととりうる値の多様性は低い．そのため，代表的な属性名（パソコンでは，「CPU」や「メモリ」「HDD」など）を思いつくことは容易である．このことから，属性名抽出の必要性は属性値抽出ほどは高くないといえる．しかし，同じパソコンであっても，「D-sub」や「電圧」など，人により必要としたり，気にしたりするかどうかが異なる特徴もある．また，「ドット抜け」や「外箱」のように，中古品として出回るときに付与される属性名もある．これらのことを考えると，やはりすべての属性名を人手で列挙することは困難である．

本研究では，ブートストラッピングによる属性名抽出手法を提案し，その基本特性の理解と，提案手法における技術的特徴の効果の検証を PC 製品のドメインにおいて行う．さらに手法の汎用性を確かめるために，その他のドメインにおいても評価実験を行う．

2 章で関連研究について，3 章でブートストラッピングを属性名に適用する際に発生する問題と問題解決にむけたアプローチについて述べる．4 章で提案手法の詳細について述べる．5 章，6 章で，PC 製品のドメインにおいて，提案手法の技術的特徴の効果の検証を行う．7 章で PC 製品以外のドメインでも動作することを確認し，手法の汎用性について示す．最後に 8 章でまとめと今後の課題について述べる．

2. 関連研究

本研究の関連研究として，ブートストラッピングによる情報抽出の研究について 2.1 節で，属性名の抽出を行う研究について 2.2 節で，その後，表を特定する研究について 2.3 節で述べる．

2.1 ブートストラッピングによる情報抽出

Riloff ら，Yangarber らは，新聞記事から人名や地名を抽出するタスクに対してブートストラッピングを用いた [9], [20]．新聞記事には決まった言い回し（人名の後ろには「氏」という単語が書かれる，など）が多く存在するため，属性値の前後のテキストをそのままテンプレートとしている．

Brin，Ciravegna らは Web 上から人名や書籍名を抽出するタスクに対してブートストラッピングを用いた [7], [8]．精度を保つために，ヒューリスティックス（著者名は大文字で始まる単語の連続とする，など）や，既存のデータベースやデジタルライブラリとの照合により抽出した値が正しいかどうかを判定している．

楠村らは，Web 文書にブートストラッピングを適用した場合の問題に対する解決方法を提案している [21]．1 つ目の問題は，表や箇条書きなどの構造化された記述形式から抽出を行うことができないということである．これに対しては，DOM 構造 [22] において上位にある語を利用したテンプレートを作成することで対応している．2 つ目の問題は，Web 上の文書の記述形式は多様であり，事例から獲得したテンプレートだけでは数が十分ではないということである．これに対しては，事例から獲得したテンプレートを交叉させ，新しいテンプレートを生成することで対応している．

本研究もブートストラッピングにより情報抽出を行うものである．上記研究との違いは，上記研究が属性値の抽出にブートストラッピングを用いているのに対して，本研究は属性名の抽出にブートストラッピングを用いる点である．3 章でも説明するが，属性名抽出に必要なテンプレートは属性値抽出に必要なテンプレートよりも多様になるため，属性名抽出の方が難しい問題といえる．

2.2 属性名の抽出

Tokunaga らは，クラス語 C を入力とし，そのクラスの属性名を抽出する手法を提案している [23]．具体的には，C を含む Web 文書を獲得し，それらからすべての名詞を属性名候補として抽出し，出現パターンと重要度によりスコアリングを行う．出現パターンとして，主に「C の A」という共起関係（A は上記で抽出した名詞）を用いている（たとえば，「机の値段」，「机の高さ」，など）．

Tengli らは，表中のセルに人手でラベル付けした事例から，表構造の特定を行っている [14]．具体的には，属性名あるいは属性値とラベル付けされた語を含むセルと隣接するセルについて，セルに含まれる文字列どうしの類似度を

計算することで，表中において属性名あるいは属性値を表すセルを特定する．また Chen らは，表中で内容の類似するセルを結合していき，結合の境界を属性名と属性値の境界と見なしている [15]．

Raghavan らはフォームなどに隣接しているテキストを属性名として抽出している [11]．He らは表の行に存在するテキストを属性名と属性値の組と見なして抽出している [12]．Zhang らは入力フォームの Widget の種類（テキスト入力フィールドやラジオボタンなど）によって抽出に用いるヒューリスティックスを変更している [13]．

Reisinger らはインスタンス（我々のキーに相当）名に相当する文字列の n-gram と属性名の候補（名詞句）の n-gram をパターンとして作成し，そのパターンの重複回数を計算することで，属性名としての妥当性を判定している [24]．

上記の研究と本研究の違いについて述べる．文献 [23] は，クラス語と共起する特定の表記パターンに基づいて属性名を抽出している．文献 [14], [15] は表構造の特定を行っている．文献 [11], [12], [13] は，表中の属性名やフォームに隣接する属性名を抽出している．しかし，これらの手法では表やフォーム，よくある言い回しといった特定の記述形式からしか属性名を抽出することができない．それに対して本研究では，表やフォーム以外にも，箇条書きや列挙など，任意の形式のテキストからも属性名の抽出が可能である．文献 [24] はインスタンス名の取得に検索エンジンのクエリログを用いている．しかし，検索エンジンのクエリを用いることができるのは，検索エンジンのサービスを行っている企業に限られる．それに対し，本研究は検索エンジンのクエリを必要とせず，任意の企業やユーザが実施できるものである．

2.3　表の特定

Web 上の表を特定する研究も多く行われている．多くの研究は HTML 上のレイアウトを目的とした表と本当の表を区別するかに焦点を当てたものである．Wang らは，機械学習ベースの手法を提案しており，表の行数と列数の平均値と標準偏差を特徴量として用いている [18]．同様の特徴量は Cafarella らの表と行の見出しを発見する研究 [16], [17] にも用いられており，Web 上の表をリレーショナルデータベースに変換することを試みている．

Yin らは，Web 上の構造化データに基づいた事実検索エンジンを開発している [19]．Yin らの事実検索エンジンでは，実体–属性–属性値の 3 つ組を Web 上の表から抽出し，ある実体の属性について尋ねるクエリに回答をしている．システムを実装するために，Web 上の属性–属性値表を特定する分類器を構築している．表を特定した後，単純に表の 1 列目を属性，表の 2 列目を属性値として抽出を行っている．

これらの研究では表を特定することに重点を置いているが，本研究では特定の記述形式を想定せず，多くの記述形式から属性名を抽出することを目指している．

表ではないが，レコード中のデータ要素（すなわち属性名と属性値）を含むページを獲得する研究もある．吉永らは，ある特定の対象物の名前を含む Web ページにおいて，データ要素を多く含むページを発見している [25]．発見手法としては，あらかじめ獲得しておいたクラスでの一般語（HTML 文書構造の情報を用い，教師なしで学習）を多く含んでいるか否かで判定している．本研究でも，データ要素を含むページを獲得するためにキーを用いているが（3.4.2 項で詳しく説明），上記研究ではユーザは調べたい対象物があらかじめ決まっている場合を対象としているのに対し，我々の研究はあるクラスのもとで，そのクラスを表現するデータ要素が書かれたページを網羅的に獲得することを目指している点に違いがある．

3.　属性名抽出におけるブートストラッピングの問題

3.1　ブートストラッピングとは

情報抽出の分野で用いられてきたブートストラッピングの目的は，ある属性名に対応する属性値の辞書を作成することである [7], [8], [9], [10]．ブートストラッピングの処理の流れは以下のようになる．まず，少数の例となる属性値（たとえば，属性名 “CPU” に対して “Core i7 3770”, “Celeron G460”）を人手で辞書に加える．次に，検索エンジンを用いて辞書中の属性値を含む Web ページを獲得する．さらに，それらのページ中でそれぞれの属性値の前後にある文字列を獲得し，[左文字列＋（抽出位置）＋右文字列] という形式のテンプレートを作成する．たとえば，**図 1** において，属性値 “core i7 3770” の前後 4 文字を用いてテンプレート [“CPU:” ＋（抽出位置）＋ “
”] を作成する．次に，これらのテンプレートをすでに獲得してあるページに対して適用することにより新しい属性値を抽出し，それらを辞書に加える．上記の処理を繰り返すことにより，少量の例から多量の属性値を獲得することができる．

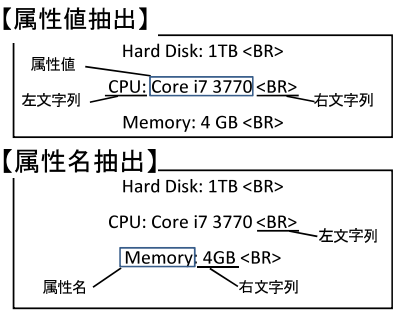


図 1　属性名抽出と属性値抽出

Fig. 1　An example of attribute value extraction and attribute name extraction.

3.2　属性名抽出に適用した際の問題

本研究では属性名の獲得にブートストラッピングを用いるが，そのまま属性名抽出に適用するには問題がある．それは，属性名の前後テキストには属性値が含まれることが多く，他の箇所に適合する確率が低いテンプレートが作成されてしまうからである．属性値抽出にブートストラッピングを適用する場合には，抽出値の前後に存在するテキストには属性名を含むことが多い．一般的に，属性名は属性値よりも多様性が低い．そのため，属性値抽出用のテンプレートは他の多くの箇所に適合すると思われる．一方，属性名の前後に存在するテキストには属性値を含むことが多い（図 1 参照）．属性値は多様性が高く，属性名抽出用のテンプレートは他の箇所に適合する確率が低くなる．その結果，Web 全体に適用したとしてもほとんど属性名が抽出できないと考えられる．また，属性値である文字列を取り除いてテンプレートを作成したとしても，そのテンプレートは一般的になりすぎて属性名以外の箇所にも適合してしまう．そのため，Web 全体に適用すると多量のノイズ（属性名以外の文字列）を抽出してしまうと考えられる．そこで本研究では，属性値である文字列を含まないようにしてテンプレートを作成し，さらにそのテンプレートの適用範囲を限定することにより上記の問題を解決することを目指す．

3.3　問題解決の着眼点

Web ページの記述形式は多様であるが，1 つのページに着目するとレコード中のデータ要素（属性名や属性値）は同じ形式で繰り返し記述されていることが多い．もし属性名抽出用のテンプレートを作成する際に属性名を 1 つだけ用いるとすると，その属性名の前後にある固定長（N 文字）の文字列をテンプレートとして獲得することになる．しかし，そのテンプレートは強制的に獲得した固定長の文字列から作成しているため，繰返しの記述形式に関係ない文字を含んでいる可能性が高い．たとえば図 1 では，属性名の前後 4 文字を用いると，“Memory” からテンプレート [“
” ＋（抽出位置）＋ “:4GB”] が作成される．文字列 “:4GB” は属性名 “Memory” に対する属性値を表す “4GB” の文字列であるため，他の属性名（“CPU” や “Hard Disk” など）を抽出するためのテンプレートとして用いることはできない．

そこで本研究では属性名抽出用のテンプレートの作成に 2 つの属性名を用いる．2 つの属性名を含むページにおいて，2 つの属性名それぞれの前後にあるテキストで完全一致する部分をテンプレートとすることで繰返しの記述形式に関係ない部分を取り除くことができる．たとえば図 1 では，“CPU” と “Memory” それぞれの前後にあるテキストで完全一致する部分からテンプレート [“
” ＋（抽出位置）＋ “:”] が作成される．

なお，属性名が 3 つ以上の場合でも提案手法は機能すると考えられる．さらに，3 つ以上とすることで属性名が記述される構造部分を限定することができるという利点も存在する．しかし，ページを検索するために正解属性名から 3 つを組み合わせると，テンプレートがさらに短くなり，ノイズを多く抽出してしまうと考えられる．そのため，構造部分を特定するために最低限必要である数として，2 つを採用した．

3.4　提案手法の工夫点

3.4.1　テンプレート適用範囲の限定

上記の方法で作成したテンプレートは HTML タグや，箇条書きを表す “-” や “*” などの記号によってのみ構成され，一般的になりすぎると考えられる．もしこのテンプレートを Web 全体に適用すると，ノイズ（属性名以外の文字列）が多く抽出されてしまう．そこで，本研究では上記の方法で作成したテンプレートを現在のページ（テンプレートを作成したページ）にのみ適用する．

しかし，同一のページでおいても上記テンプレートはページ中の多くの箇所に適合し，属性名以外の文字列を抽出してしまうと考えられる．この問題を解決するために，テンプレートの適用範囲を 2 つの属性名が記述された文書範囲に限定する．この文書範囲は，そのページの文書構造を解析することにより特定する．

3.4.2　ノイズとなるページの回避

また，2 つの属性名で Web を検索した場合，レコード中のデータ要素を含まないページ（Wikipedia[*1]のような用語解説ページや，Yahoo!知恵袋[*2]のような掲示板など）が獲得されるという問題がある．たとえば，「ハードディスク メモリ」というクエリを用いて検索した場合，獲得される掲示板のページとしては，「いろいろ調べてみましたがハードディスク容量，メモリとも問題なさそうでした.」という Q&A に関するページがあげられる．これらのページはクエリとして用いた属性名以外の属性名を含んでいないことが多く，これらのページから他の属性名を抽出することはできない．

そこで本研究では，2 つの属性名に加えてキーを用いて Web を検索する．キーとはオブジェクトを一意に特定することができる値のことで，「パソコン」の場合は，機種名か製品名に相当する．これにより，他の属性名を含んでいないページが獲得されるのを防ぐ．

オブジェクトの名称となる文字列を含んでいるページのみを属性名抽出の対象にするという考え方は，Reisinger らによる属性名の抽出手法でも実現されている [24]．また，吉永らは，属性名が書かれているページを発見する手法を提案しているが，彼らもその手がかりとしてオブジェク

*1　http://ja.wikipedia.org/wiki/メインページ
*2　http://chiebukuro.yahoo.co.jp/

ト名を用いている [25]．したがって，オブジェクト名を手がかりとすることそのものには新規性はない．ただし，決まったキーのみを用いていると，多様な属性名が抽出できない恐れがある．たとえば，ある Web サイトではメーカごとに性能比較表を提供しているかもしれないし，別の Web サイトでは現行機種の情報のみを提供しているかもしれない．そこで我々は属性名の辞書だけでなくキーの辞書もブートストラッピングで学習する．これにより，属性名の抽出対象となるページの種類を増やすことができると思われる．

3.5　技術的特徴のまとめ

提案手法の技術的特徴は以下のとおりである．

- 2 つの属性名を含むページにおいて，2 つの属性名それぞれの前後にあるテキストで完全一致する部分を属性名抽出用のテンプレートとする．これにより，属性値を含めずにテンプレートを作成できるうえ，ページごとに異なる属性名の記述形式に対応できる．
- 上記テンプレートは，そのテンプレートを作成したページにのみ適用する．さらに，文書構造を参照することで，テンプレートの適用範囲を限定する．これにより，属性名が記述された範囲以外の部分にテンプレートを適用することで属性名以外の語を抽出してしまうことを防ぐ．
- 属性名を獲得するブートストラッピングに加えて，属性名抽出の手がかりとなるキーを獲得するブートストラッピングも実行する．これにより，不要な Web ページの検索を防ぎ，さらに多様な属性名の獲得を目指す．

4.　ブートストラッピングによる属性名抽出

提案する属性名抽出手法の流れを**図 2** に示す．以下，4.1 節，4.2 節でキー抽出の手順，属性名抽出の手順についてそれぞれ述べる．4.3 節では上記の手順中で用いる評価方法（Yangarber らの方法 [20]）について述べる．

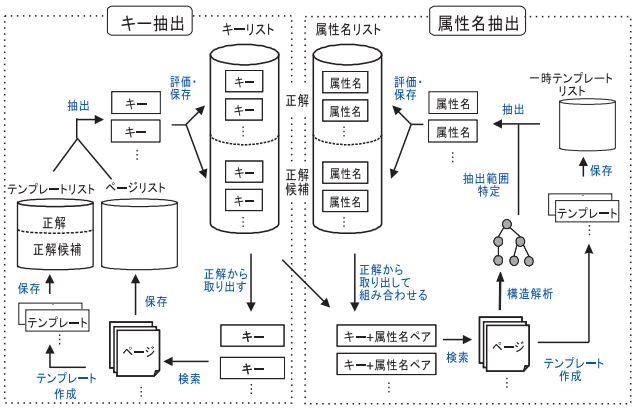


図 2　処理の流れ

Fig. 2　Flow of our proposed method.

4.1　キー抽出の手順

キーを抽出するブートストラッピングではあらかじめ以下のリストを用意しておく（図 2 左側）．

- キーリスト：キーを正解と正解候補に分けて保存しておくリスト
- テンプレートリスト：テンプレートを正解と正解候補に分けて保存しておくリスト
- ページリスト：Web ページを保存しておくリスト

本研究では，ブートストラッピングにおける一連の値抽出のプロセスをサイクルと呼ぶ．通常ブートストラッピングでは，このプロセスを繰り返すことで値を抽出する．以下の手順では，Step 1.〜Step 6. がサイクルとなる．ページリストでは，1 つのサイクルが終了してもリストは初期化せず，次のサイクル以降でも続けて利用する．

キー抽出は以下の手順で行う（図 2 左側）．あらかじめ人手で少数の例となるキーをキーリストに正解として加えておく．

Step 1.　キーリストから，これまでのサイクルでクエリとして使用していない正解キーをすべて取り出す．

Step 2.　Step 1. で取り出した正解キーについて，1 つ 1 つをクエリとして Web を検索して適合したページを獲得し，それらをすべてページリストに保存する．

Step 3.　Step 2. で獲得したすべてのページにおいて，そのページを獲得するにあたってクエリとして用いたキーの前後 N 単語をテンプレートとして獲得し，テンプレートリストに正解候補として保存する．このとき，HTML タグも 1 単語と見なす．

Step 4.　テンプレートリスト中の全テンプレート（正解，正解候補の両方）を用いてページリスト中の全ページからキーを抽出する．ただし，各テンプレートは，これまでのサイクルで適用していないページにのみ適用する．このとき，抽出されたキーについて，すでに正解としてキーリストに含まれているか否か，正解候補としてキーリストに含まれているか否かを確認し，その情報をテンプレートに付与する．また，正解・正解候補のいずれでもない場合（現在のサイクルで初めて抽出されたキーである場合）には，そのキーを正解候補として新たにキーリストに保存する．

Step 5.　すべての正解候補テンプレートについて，Step 4. で付与された情報（抽出したキーが正解である/ない）から評価値（Yangarber らの確信度 [20]）を求め，上位 h 個を正解としてテンプレートリストに保存しなおす（評価値についての詳細は 4.3 節で述べる）．このとき，テンプレートにはそのテンプレートが以前のサイクルで抽出したキーに関する情報も付与されており，それらも評価値の算出に用いられることとなる．

Step 6.　すべての正解候補キーについて，これまでのサイクルでそのキーを抽出した正解テンプレートに付与

された情報から評価値を求め，上位 i 個を正解としてキーリストに保存する．

Step 7.　Step 1. に戻る．

4.2　属性名抽出の手順

属性名を抽出するブートストラッピングではあらかじめ以下のリストを用意しておく（図 2 右側）．

- キーリスト：キーを正解と正解候補に分けて保存しておくリスト（キー抽出（4.1 節）と共通）
- 属性名リスト：属性名を正解と正解候補に分けて保存しておくリスト
- 一時テンプレートリスト：一時的にテンプレートを保存しておくリスト

属性名抽出は以下の手順で行う（図 2 右側）．あらかじめ人手で少数の例となる属性名を属性名リストに正解として加えておく．また，正解キーも先の 4.1 節の手順においてキーリスト中に保存されているものとする．また，サイクルとは，属性名抽出では以下の Step 1.〜Step 5 までの手順を指す．

Step 1.　キーリストと属性名リストから，これまでのサイクルでクエリとして使用していない [正解キー，正解属性名 A，正解属性名 B] の組合せ（以下，キー＋属性名ペア，と呼ぶ．正解属性名 A と正解属性名 B は異なる属性名である）をすべて作成して取り出す．

Step 2.　Step 1. で取り出したキー＋属性名ペアについて，1 つ 1 つをクエリとして Web を検索し，適合したページをすべて獲得する．

Step 3.　Step 2. で獲得した各ページについて，以下の Step 3-1. と Step3-2. を実行する．

Step 3-1.　ページを獲得するにあたってクエリとして用いたキー＋属性名ペアに含まれる 2 つの属性名の前後にあるテキストの完全一致部分からテンプレートを作成する．さらに，DOM 構造 [22] を参照することによりテンプレートを適用する範囲を特定する（それぞれ，詳細については後述）．

テンプレートの作成と適用範囲の特定ができた場合は，作成したテンプレートを一時テンプレートリストに保存し Step 3-2. へ進む．できなかった場合はそこで終了する．

Step 3-2.　上記テンプレートを用いて上記適用範囲から属性名を抽出する．このとき，抽出された属性名について，すでに正解として属性名リストに含まれているか否か，正解候補として属性名リストに含まれているか否かを確認し，その情報をテンプレートに付与する．また，正解・正解候補のいずれでもない場合（現在のサイクルで初めて抽出された属性名である場合）には，この属性名は正解候補として新たに属性名リストに保存する．

Step 4.　一時テンプレートリスト中の全テンプレートについて，Step 3-2. で付与された情報（抽出した属性名が正解である/ない）からテンプレートの評価値（Yangarber らの確信度 [20]）を求め，上位 j 個を正解とする（評価値についての詳細は 4.3 節で述べる）．

Step 5.　正解候補属性名について，その属性名を抽出した正解テンプレートに付与された情報から評価値を求め，上位 k 個を正解として属性名リストに保存しなおす．属性名抽出ではテンプレートをそのテンプレートを作成したページにのみ適用し，後のサイクルでは用いないため，一時テンプレートリスト中のテンプレートはすべて破棄する．

Step 6.　Step 1. に戻る．

Step 3-1. のテンプレートの作成の詳細な手順は以下のとおりである．2 つの属性名（属性名 A，B とする）に対して前後テキストの完全一致部分をテンプレートとする．この完全一致部分は，属性名 A，B それぞれの前方あるいは後方に 1 文字ずつ見ていき，一致すればその文字列をテンプレートに含めて次の文字に進み，一致しなければそこで終了という方法で特定する．ここでテンプレートを作成する基となった文字列は属性名 A の前後（テンプレート文字列 A とする）と属性名 B の前後（テンプレート文字列 B とする）に存在することとなる．

Step 3-1. の適応範囲の特定の詳細な手順は以下のとおりである．

Step 3-1-1.　DOM 構造において，テンプレート文字列 A，B をそれぞれ含むような最小の部分木を 2 つ特定する．ここでそれら部分木の根ノードをそれぞれノード A，B と呼ぶ．さらにノード A，B 自身とそのすべての子孫からなる部分木をそれぞれ部分木 A，B と呼ぶ（**図 3** 参照）．

Step 3-1-2.　ノード A，B が一致すれば，それ自身とすべての子孫からなる部分木に該当するテキストをテンプレートの適用範囲とし，Step 3-2. へ進む．一致しなければ Step 3-1-3. へ進む．

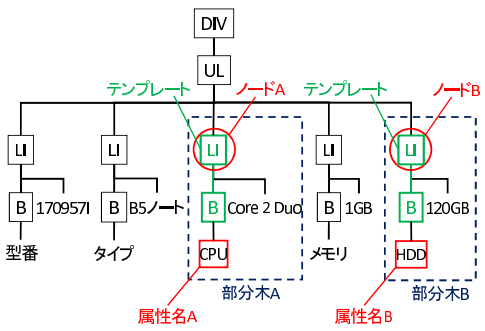


図 3　DOM 構造の例

Fig. 3　An example of DOM structure.

Step 3-1-3. ノード A，B の親ノードをそれぞれ特定する．2 つの親ノードが一致すれば，それ自身とすべての子孫からなる部分木に該当するテキストをテンプレートの適用範囲とし，Step 3-2. へ進む．一致しなければ終了する．

4.3 Yangarber らによる確信度

ブートストラッピングでは誤った値（キー抽出ならキー，属性名抽出なら属性名）がリストに混入すると，その値によって誤ったテンプレートが作成され，それによりさらに誤った値が抽出されてしまう．これを防ぐためには，サイクルごとに抽出値とテンプレートを再評価する必要がある．本研究では，Yangarber ら [20] が用いた確信度を評価値として利用し，上位のもののみを正解とすることで精度を保つ．Yangarber らは，ブートストラッピングにおいてテンプレート t_i に対する確信度 $C_{template}(t_i)$ と，抽出値 v_j に対する確信度 $C_{value}(v_j)$ を次のように計算している．

$$C_{template}(t_i) = conf(t_i) \cdot \log |pos(t_i)| \quad (1)$$

$$C_{value}(v_j) = 1 - \prod_{t_k \subset M_t} (1 - conf(t_k)) \quad (2)$$

- $conf(t_i)$：t_i が抽出した値について，すべての抽出値の数に対する正解である抽出値の数の割合
- $pos(t_i)$：t_i が抽出した正解抽出値の数
- M_t：v_j を抽出した正解テンプレートの集合

とする．

本研究では，$C_{value}(v_j)$ をキーと属性名のそれぞれの評価値，$C_{template}(t_i)$ をテンプレートの評価値として用いる．

5. キー抽出実験

提案手法を実行するためには，まずは適切にキーが抽出されなければならない．本章では，キー抽出のブートストラッピングにおいてどのパラメータの組合せの場合に最良の結果を出すかを調査する．キー抽出のブートストラッピングにおけるパラメータには以下のものがある．以降，これらのパラメータをキー抽出パラメータと呼ぶ．

- テンプレート単語数：テンプレートに用いる単語の数（4.1 節 Step 3. の N）
- テンプレート更新数：各サイクルで正解として保存するテンプレートの数（4.1 節 Step 5. の h）
- キー更新数：各サイクルで正解として保存するキーの数（4.1 節 Step 6. の i）

実験条件を以下に示す．ブートストラッピングはテンプレートをページにあてはめることにより属性値を抽出する手法であり，作成されるテンプレートの数が増えるにつれて，計算時間は徐々に増えていく．よって，多数の精度調査を効率的に行うために，取得するページ数が 5 万を超えた時点で実験を終了することにする．本実験でページ収集する際には，Yahoo! JAPAN によって提供されている Web 検索 API を用いることにする．本実験では，データを抽出する対象ドメインとして PC 製品を用いることにする．

MSI Wind
MSI Wind U135DX
Lenovo ThinkPad
Lenovo ThinkPad X200
HP TouchSmart
HP TouchSmart 310
Fujitsu Lifebook
Fujitsu Lifebook t1010
Acer Aspire
Acer Aspire 5742

図 4　初期の正解キー
Fig. 4　Correct keys for initial seeds.

キー抽出のブートストラッピングでは正解となるキーを人手で少数与えておく．そこで，キー抽出実験の前に，5 名の学生にキーとなる PC 製品の名前を 10 個探すというタスクを課した．我々は計 50 個からランダムで 5 つ選択し，各キーに対して 2 種類の表現を用いることにした．2 種類の表現とは，PC のシリーズ名と製品名を指す．たとえば，シリーズ名は「Lenovo ThinkPad」で，製品名は「Lenovo ThinkPad X200」である．前半のキーは，“ThinkPad” に関するページを取得するのに役立つ．後半のキーは，このキーから作成されるテンプレートをページにあてはめることで，“ThinkPad X200” のような製品名を抽出するのに役立つ．また，後半のキーから作成されるテンプレートは他社の PC 製品名も取得できると考えられる．本実験で用いた初期の正解キーを**図 4** に示す．我々はキー抽出パラメータに初期値を設けた．テンプレート単語数の初期値は 2，キー更新数の初期値は 10，テンプレート更新数の初期値は 50 とした．1 つのパラメータを変化させて実験をする際，その他のパラメータは初期値をとるものとする．テンプレート単語数は 1〜4 の範囲で，キー更新数は 2〜20 の範囲で，テンプレート更新数は 20〜200 の範囲で調査した．

我々は抽出された値が「（メーカ名）＋（シリーズ名）＋（製品番号）」，「（メーカ名）＋（製品番号）」，「（シリーズ名）＋（製品番号）」であれば，正解と見なす．キー評価での精度は以下のように算出した．本実験で抽出した正解キーは属性名抽出実験で用いられる．

（精度）＝（人手でキーと判断した数/抽出数）　(3)

5.1 キー抽出実験の結果

図 5 はテンプレート単語数を変化させた場合の精度の推移を示している．抽出数が 80 までは精度はテンプレート単語数が 4 の方が良く，5 万ページ取得時には精度はテンプレート単語数が 2 の方が良い．**図 6** はキー更新数を変化させた場合の精度の推移を示している．サイクルの前半ではキー更新数が 5 の精度が最も良く，サイクルの後半ではキー更新数が 10 の精度が最も良い．**図 7** はテンプレート

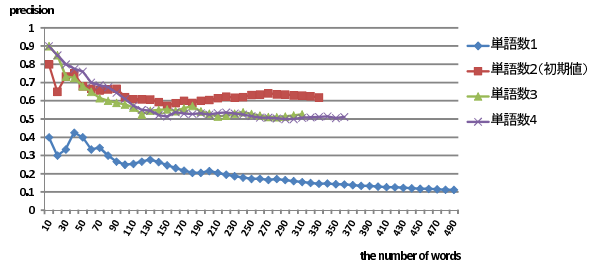


図 5　各テンプレート単語数における精度の推移

Fig. 5　Transition of precision according to the number of extracted keys in each fixed number of words in template.

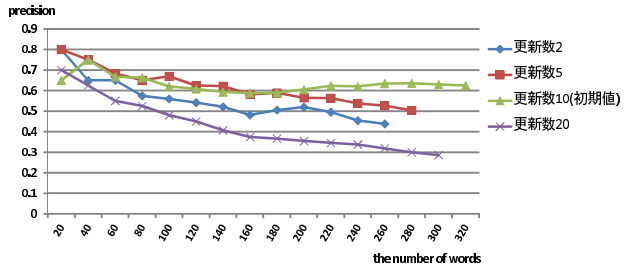


図 6　各キー更新数における精度の推移

Fig. 6　Transition of precision according to the number of extracted keys in each fixed update number of keys.

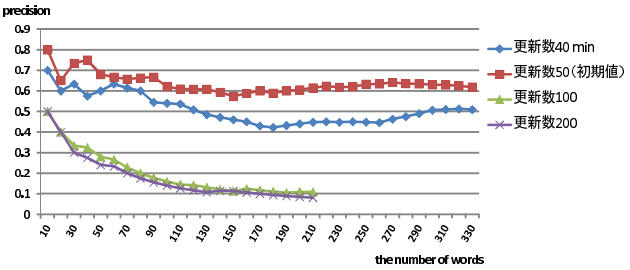


図 7　各テンプレート更新数における精度の推移

Fig. 7　Transition of precision according to the number of extracted keys in each fixed update number of templates.

更新数を変化させた場合の精度の推移を示している．全サイクルにおいて，テンプレート更新数が 50 の精度が最も良い．正解キーの抽出数（精度 × 抽出数）をパラメータ選択の基準とする．すると PC 製品のドメインにおいてキー抽出パラメータの最良な組は（テンプレート単語数，キー更新数，テンプレート更新数）が（2，10，50）であった．

6.　属性名抽出実験

本章では属性名抽出手法の有効性の評価を行う．具体的には，我々の提案した技術的特徴がどれほど精度・抽出数に影響するかを調査する．また，属性名抽出実験においても，最良の組合せとなるパラメータの組を調査する．

6.1　実験方法

提案手法での技術的特徴の有効性を評価するため，4 つの手法を設定する．1 つ目は “提案手法”，2 つ目は “構造参照なし” 手法，3 つ目は “キー固定” 手法，4 つ目は “キーなし” 手法である．提案手法では，属性名を 2 つ使用することと，キーを多様化すること（ブートストラッピングを用いてキーを増やすこと）と，文書構造を参照することを行う．構造参照なし手法では，属性名を 2 つ使用することと，キーを多様化することを行う．キー固定手法では，属性名を 2 つ使用することと，キーを使用することと，文書構造を参照することを行う．キーなし手法では，属性名を 2 つ使用することと，文書構造を参照することを行う．以上の手法と技術的特徴の有無の関係を**表 1** にまとめる．4 手法を比較することにより，各技術的特徴がどれほど精度・抽出数に影響するかを調査する．本実験でページ収集する際にも，Yahoo! JAPAN によって提供されている Web 検索 API を用いることにする．また属性名抽出実験も 5 万ページ取得時に実験を終了する．

表 1　技術的特徴の比較

Table 1　Comparison of technical features.

	属性名 2 つ使用	キー使用	キー多様化	文書構造参照
キーなし	○			○
キー固定	○	○		○
構造参照なし	○	○	○	
提案手法	○	○	○	○

cpu
os
memory
hdd
processor
display size
weight
ram
brand
color

図 8　初期の正解属性名

Fig. 8　Correct attribute names for initial seeds.

属性名のブートストラッピングでは正解となる属性名を人手で少数与えておく．そこで，属性名抽出実験の前に，8 名の学生に PC 製品の属性名を 10 個あげるというタスクを課した．我々は計 80 個から，数の多い属性名の上位 10 種を選択し，初期サイクルの正解属性名とした．本実験で用いた初期サイクルの正解属性名を**図 8** に示す．また，属性名抽出をするブートストラッピングでは，クエリ作成のためにキーを用いる．我々はキー抽出実験で得た正解キーを用いた．本実験では，キー抽出実験において各サイクルで抽出した正解キーを，同じサイクルのときに正解キーを与えることにした（たとえば，2 サイクル目で得られた正解キーを，そのまま属性名抽出実験時の 2 サイクル目で与える）．

属性名抽出のブートストラッピングにおけるパラメータには以下のものがある．以降，これらのパラメータを属性

名抽出パラメータと呼ぶ．

- テンプレート更新数：各サイクルで正解として保存するテンプレートの数（4.2 節 Step 4. の j）
- 属性名更新数：各サイクルで正解として保存する属性名の数（4.2 節 Step 5. の k）

以下，属性名抽出パラメータの組を（属性名更新数，テンプレート更新数）で表すことにする．我々は以下の範囲で最良の属性名抽出パラメータの組を調査した．キーなし手法では（30, 140）という初期値で，属性名更新数は 20〜30，テンプレート更新数は 120〜500 という範囲で調査した．キー固定手法では（30, 120）という初期値で，属性名更新数は 20〜50，テンプレート更新数は 60〜140 という範囲で調査した．構造参照なし手法では（30, 100）という初期値で，属性名更新数は 30〜90，テンプレート更新数は 60〜140 という範囲で調査した．提案手法では（30, 120）という初期値で，属性名更新数は 20〜50，テンプレート更新数は 60〜140 という範囲で調査した．1 つのパラメータを変化させて実験を行い，最適なパラメータの組を決定する．

属性名の評価に関しては，実際にその属性名に属性値を持つかどうかを人手で Web ページを調べて判断した．精度は以下のように算出した．

（精度）＝（人手で属性名と判断した数/抽出数）　(4)

6.2　結果と考察

提案手法における各属性名抽出パラメータでの実験結果を以下に載せる．**図 9** は属性名更新数を変化させたときの精度の推移を示している．**図 10** はテンプレート更新数を変化させたときの精度の推移を示している．各プロットは各サイクルの値を示す．たとえば，プロットが 2 つあった場合，2 回のサイクルが行われたことになる．他 3 手法も同様に，パラメータの組と精度の推移の関係を調査した．各手法間の比較を行うため，正解抽出数が最も多いパラメータをそれぞれ選択した．各手法の精度の推移や用いた属性名抽出パラメータの組を**図 11** に載せる．また，各手法における精度や正解抽出数，全体の抽出数を**表 2** にまとめた．

図 9 より，属性名更新数を増やせば増やすほど，抽出数が増大する一方，精度は低下する．ここで，抽出数とは各サイクルで正解候補から正解とする属性名の個数を指すが，属性名更新数が大きいとき，最大の個数分抽出しきれていないことが分かる（たとえば，属性名抽出パラメータが (40, 120) のとき，2 サイクル目の最大の抽出数は 40×2 の 80 である）．この理由について説明する．属性名のブートストラッピングでは，正解属性名から 2 つ，正解キーから 1 つ取り出し，クエリとして使用する．2 サイクル目でクエリの候補は数十万もの数にのぼっていたが，今回は 5 万ページを超えたサイクルで終了しているため，すべてのクエリの候補を用いていない可能性がある．まとめると，グラフの傾きに大差はなく，属性名更新数が大きい場合は抽出数が伸びる余地が存在する．

図 10 より，全体の傾向として，テンプレート更新数を増やせば増やすほど，精度は良くなる．これは我々の提案するテンプレート作成方法や適用方法がノイズを抽出しにくいものであるからと考えられる．以降の，正解キーを各サイクルごとに与える場合の提案手法では（属性名更新数，テンプレート更新数）が（40, 120）のパラメータを用いる．

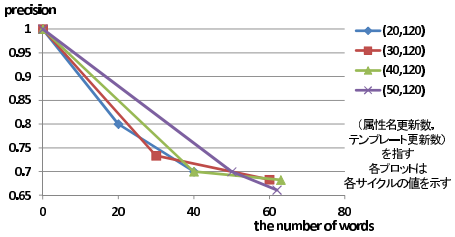


図 9　提案手法で，属性名更新数と精度・抽出数との関係
Fig. 9　The precision in each cycle in each fixed number of update number of attribute names in the proposed method.

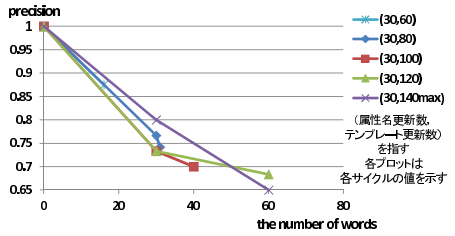


図 10　提案手法で，テンプレート更新数と精度・抽出数との関係
Fig. 10　The precision in each cycle in each fixed number of update number of templates in the proposed method.

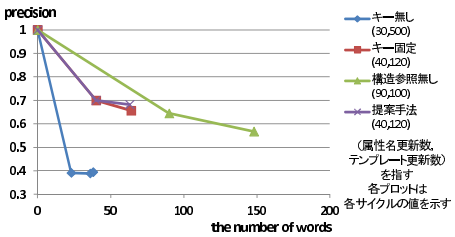


図 11　各技術的特徴の効果
Fig. 11　Effects of technical features.

表 2　各技術的特徴の評価
Table 2　Evaluation of technical features.

	精度	正解抽出数（全抽出数）
キーなし	0.395	15（38）
キー固定	0.656	42（64）
構造参照なし	0.568	84（148）
提案手法	0.683	43（63）

表 3　キーの設定方法の違い
Table 3　Precision and the number of extracted attribute names for each initial key settings.

	精度	正解抽出数（全抽出数）
提案手法（正解キーを各サイクルごとに与える）	0.683	43（63）
提案手法（正解キーを初期サイクルで与える）	0.691	152（220）

6.3　技術的特徴の効果

手法間の差について考察する（図 11 参照）．まずは，構造参照なし手法と提案手法とを比較する．正解抽出数は構造参照なし手法の方が多い．また，精度に関しては，抽出数が少ないので構造参照なしより高い精度であるかどうかは明らかではない．次に，キー固定手法と提案手法とを比較する．精度・正解抽出数ともにほとんど同じであった．差がつかなかった理由について 6.4 節で考察する．最後に，キーなし手法と提案手法とを比較する．精度・正解抽出数ともに提案手法が良い．パソコンの機種名，製品名をキーとして使用することで，検索をパソコンのページに限定することができたことによると考えられる．

文書構造を参照することの有効性は，属性名の抽出数という観点では見い出すことができなかった．そこで，もう少し詳細に文書構造を参照することの影響を調査するため，我々は構造参照なし手法にのみ現れる「正解でない」属性名を調査した．すると，企業情報を示す「about us，contact us，privacy policy，sitemap」のような単語や，ショッピングサイトでの機能を表す「cart」，レビューサイトへのリンクを表す「reviews」，他製品情報サイトへのリンクを表す「printer，ink，monitor」のような単語が含まれていた．これらの単語は，パソコンの製品属性が書いてある部分とは離れた場所に書いてあることが多く，提案手法でこれらを抽出しなかったことについては，構造参照を行った効果が表れているといえる．

また，構造参照まで考慮することで得た知見もある．オブジェクトの属性情報は表に書かれていることがあるが，我々は当初，オブジェクトの属性情報が書かれている表は，1 ページ中に 1 つしかないと考えていた．しかし実際には，複数の表に属性情報が書かれている場合があることを発見した．さらに，そのような場合は 2 種類あることが確認できた．1 種類目はたとえば，1 つの製品の性能について書かれた表でも CPU，メモリのような性能の情報を示す表構造と，色や大きさのような外形の情報を示す表構造に分けられる場合である．2 種類目は 1 ページ中に複数の製品の情報が存在しており，なおかつそれらが別々の表に書かれている場合である．構造参照した場合は，前者の場合は，明らかにもう一方の表に入っている属性はとることができない．後者の場合も，別の製品の表に，その製品特有の属性が入っている場合には，それをとることができない（たとえば，DVD ドライブの有無や付属ソフト）．

これらのことより，文書構造の参照を行うことにより，PC 製品に関係のないキーワードを取得せずにすんだという利点がある一方，抽出箇所を限定してしまうことにより取得できなかった属性名もあることが分かる．提案手法では，テンプレート作成に用いた文字列を含む文書構造部分をボトムアップに調べる方法をとったが，文書部分を限定するか否かを文書全体を見て判断するような工夫を行う必要があると考える．

6.4　キーの設定

本実験においてキーは，キー抽出実験で得られた正解キーをそのままのサイクルの順序で与えていた．しかし，今回の実験は 2 サイクルの処理であったので，キー固定手法と提案手法との違いは，2 サイクル目に追加したキーのみとなる．その結果，正解抽出数に差がつかなかったと考えられる．そこで，我々は初期サイクルのキーの正解リストに，キー抽出実験で得られた正解キーをすべて与える手法を試みた．**表 3** に正解キーを各サイクルで与えた場合と初期サイクルで与えた場合との精度・正解抽出数を載せる．正解キーを各サイクルで与える手法よりも，初期サイクルで与える場合の方が 100 個以上正解抽出数が増え，さらに精度も若干上昇している．

6.5　再現率の評価

ここまでの評価実験では，属性名抽出の有効性の評価を精度と抽出数により行ってきた．事前に正解データとして抽出すべき属性名を網羅的に与えることができれば，再現率も評価することができる．再現率を評価できれば，手法の有効性をより的確に評価できる．しかし，人手で網羅的に正解となる属性名を与えることは難しい．そこで，本研究では限られた範囲内ではあるが，複数人の被験者により限られた条件下で列挙された属性名を近似正解集合と見なし，これを用いて再現率を算出することにした．

我々は，近似正解集合を獲得するために被験者実験を行った．近似正解集合として，ユーザが何も見ずにあげた属性名と，ユーザが Web を見て発見した属性名の 2 種類を用意した．前者は，被験者に時間無制限で考えつかなくなるまで，何も見ずに属性名をあげてもらうことで獲得する．これは，誰もが容易に思いつく重要な属性名をどれだけ網羅できるかを調べるために設定した．後者は，被験者に時間制限内で，Web を自由に閲覧してもらって発見した

属性名をあげてもらうことで獲得する．これは，前者ほど重要ではなくても，言われれば属性名となるものまで網羅できるか調べるために設定した．

再現率の算出には以下の式を用いた．

$$（再現率）= |A \cap B|/|B| \quad (5)$$

ただし，A はシステムが抽出した属性名集合，B は近似正解集合（ユーザが何も見ずにあげた属性名の集合，またはユーザが Web を見て発見した属性名の集合）を意味する．

実験には日頃からプログラミングを行っている 6 名の被験者（うち 4 名は PC を自作した経験がある人）に参加してもらった．これまでの評価実験で得た結果の中では，キーのブートストラッピングで得た値を最初からすべて与える場合が，最も抽出数と精度も高かった（表 3 参照）．そこで，同じ条件で抽出した属性名の中で，上記近似正解集合に含まれるものがどれだけあるかで，再現率を計算した．

その結果，ユーザが何も見ずにあげた属性名を近似正解集合とした場合の再現率は 0.31，ユーザが Web を見て発見した属性名を近似正解集合とした場合の再現率は 0.20 となった．

再現率の数値そのものを見ると，良い結果であったとはいえない．再現率が低くなった理由を調べるために，近似正解集合中の属性名と，システムが抽出した属性名を調査した．すると両方に，かなりの言い換えが含まれていた．たとえば，バッテリに対しては，“battery”，“batteries”，“battery life” が，CPU に対しては，“CPU”，“CPU type”，“CPU spee” が近似正解集合に含まれていた．また，システムは同じスクリーンに関する属性名でも，“screen”，“screen size”，“screen resolution”，“max screen resolution”，“screen size (measured diagonally)” を抽出していた．このように，両者が抽出したものは言い換えを含めると多様になる．また，システムは “voltage required”，“max transfer rate”，“microsoft office preloaded” など，ユーザが気づきにくい属性も数多く抽出していた．これらの点から，両者のマッチングをとると，再現率が低くなってしまったものと思われる．また，本論文では属性名抽出実験を 5 万ページで限定しているので，引き続きページを取得していけば，さらに再現率が上昇する可能性がある．

7. 他ドメインへの適用

ここまでのキー抽出実験と属性名抽出実験は，提案手法の基本特性を理解することと，手法中の技術的特徴の効果を検証するために，PC 製品のドメイン（以降，PC ドメイン）を対象に行ってきた．しかし，手法の汎用性を示すためには，1 つのドメインのみの効果を示すだけでは不十分である．そこで本章では，PC 製品以外のドメインにも適用し，手法の汎用性について検証する．また，これらのドメインで得られた実験結果を，2 章の関連研究の論文で報告されている実験結果と比較し，従来手法との比較を行う．PC 製品は電気製品のカテゴリに属すると考えらえれるが，これとは異なるカテゴリからドメインを選択する．具体的には，機械製品のカテゴリから自動車製品を，食品カテゴリからドリンク製品（栄養ドリンクとスポーツドリンク）を実験の対象とする（以降，自動車ドメイン，ドリンクドメインと呼ぶ）．

7.1 キー抽出実験

7.1.1 実験方法

自動車ドメインとドリンクドメインにおけるキー抽出実験の方法は，5 章で PC 製品に対して行ったものと同様である．ブートストラッピングで必要な初期の正解キーの獲得も，PC ドメインのときと同様，被験者 8 名にキーとなる自動車製品（またはドリンク製品）の名前を 10 個探すタスクを与えることで獲得した．実際には，獲得した 80 個からランダムに 5 つ選択し，各キーに対してシリーズ名と製品名の 2 種類を用意した．たとえば，自動車製品では，シリーズ名「Mazda」に対し，「Mazada CX-9」が製品名となる．ドリンク製品では，シリーズ名「Emergen-C」に対し，「Emsergen-C Raspberry」が製品名となる．本実験において各ドメインで用いた初期の正解キーは**表 4** のようになる．

キー抽出用のパラメータを設定するために，自動車ドメインでは，テンプレート単語数は 2〜4 の範囲で（PC 製品では 1 の場合は明らかに悪かったので省略），キー更新数は 5〜20 の範囲で，テンプレート更新数は 100〜300 の範囲で調査した．それぞれ初期値は，2，10，200 である．ドリンクドメインでは，テンプレート単語数は 2〜4 の範囲で，キー更新数は 5〜20 の範囲で，テンプレート更新数は 20〜200 の範囲で調査した．それぞれ初期値は，2，10，50 である．また，キー抽出実験は 2.5 万ページで終了した．

7.1.2 実験結果

自動車ドメインにおける，パラメータを変えた際の抽出結果を**図 12**，**図 13**，**図 14** に示す．図 12 はテンプレー

表 4　各ドメインで用いた初期サイクルの正解キー
Table 4　Keys used as initial seeds for each domain.

自動車ドメイン	ドリンクドメイン
Jeep	Red Bull
Jeep Liberty	Red Bull Sugarfree
Mazda	Zipfizz
Mazda CX-9	Zipfizz Healthy Energy Drink Mix
Ford	Emergen-C
Ford Fusion	Emergen-C Raspberry
Nissan	Monster Energy
Nissan Cube	Monster Energy Absolutely Zero
Toyota	5 Hour Energy
Toyota Camry	5 Hour Energy Grape

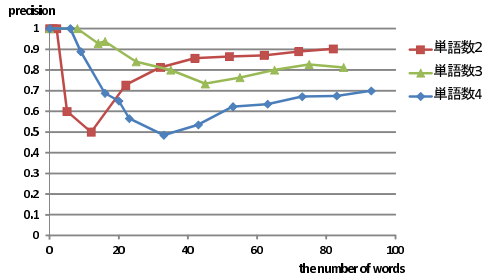


図 12　各テンプレート単語数における精度の推移

Fig. 12　Transition of precision according to the number of extracted keys in each fixed number of words in template (Autos domain).

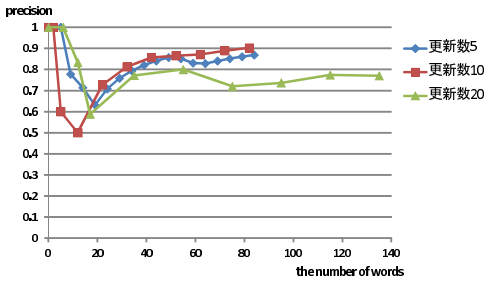


図 13　各キー更新数における精度の推移

Fig. 13　Transition of precision according to the number of extracted keys in each fixed update number of keys (Autos domain).

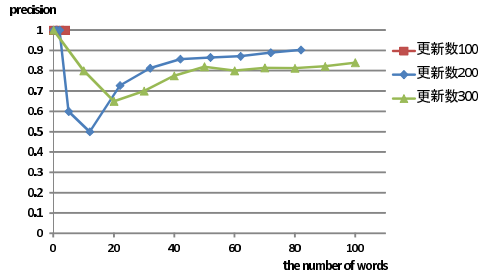


図 14　各テンプレート更新数における精度の推移

Fig. 14　Transition of precision according to the number of extracted keys in each fixed update number of templates (Autos domain).

ト単語数を変化させた場合，図 13 はキー更新数を変化させた場合，図 14 はテンプレート更新数を変化させた場合のグラフである．自動車ドメインでは，初期のサイクル数で精度が極端に減少し，その後上昇するという傾向が見られた．これは，サイクルの初期ではキーの抽出数が小さく，確信度の計算が正確に行えていないためである．その後，抽出数が大きくなると，確信度の評価の効果が表れ，精度が上昇したものと思われる．正解キーの抽出数（精度 × 抽出数）をパラメータ選択の基準とする．するとテンプレート単語数は，PC ドメインの場合と同様，単語数が 2 の場合が最も良くなった．キー更新数は 20 のとき，テンプレート更新数は 200 のときが最も良くなった．以降の属性名抽出実験では，これらの値を用いる．

一方，ドリンクドメインでは，いずれのパラメータを用いても，初期サイクルの時点で誤った値を抽出してしまい，正しいキーはほとんど抽出できなかった．ドリンク製品は，医療系のコンテンツと深い結び付きを持っているため，ドリンクの製品名がその他の医療サービスと同列に扱われている Web ページがあった．初期サイクルで新たに取得した値を見てみると，“healthsouth rehabilitation hospital” や “rehabilitation aide”，“vocational rehabilitation” などの，リハビリ関連のサービスや病院に関するキーワードが多く抽出されていた．また，“Career Medical Training”，“x-ray technician radiology jobs”，“rehabilitation services technician”，“veterinary technician” など，医療系の求人に関するキーワードも多く取得されていた．これ以降のサイクルでもこれらのキーワードばかりが抽出されてしまった．これは，健康飲料の摂取は，健康管理における 1 つの処方として考えられており，健康管理に関連するリハビリテーションが同等の値として取り上げられたためだと考えられる．また，飲料の製造企業がこれらの求人を含む医療系サービスも提供しており，求人サービスのページが多くヒットしてしまったからだと思われる．

これらのことから，多様なキーが精度良く抽出できるかどうかは，ドメインに依存することが分かる．また，ブートストラッピングが正しく動作しないドメイン（上記のドリンクドメイン）では，サイクルの初期のころに多くのノイズとなるキーワードが入ってしまい，正しいキーがまったく取得できないことが分かる．

7.2　属性名抽出実験

7.2.1　実験方法

自動車ドメインとドリンクドメインにおける属性名抽出実験の方法は，6 章で PC ドメインに対して行ったものと同様である．自動車製品の属性名は，“Displacement（排気量）” や，“Seating capacity（座席定員）”，“Length（車長）” のように性能や外形を示す単語と，ドリンク製品の属性名は，“Protein（タンパク質）” や，“Vitamin C（ビタミン C）”，“weight（重量）” のように成分量や内容量を指す単語とする．

各ドメインの初期サイクルの正解属性名を決定するため，PC ドメインと同様に，被験者 8 名に属性名をあげるタスクを課した．我々は数の多い属性名の上位 10 種を選択し，正解属性名とした．各ドメインで用いた初期サイクルの正解属性名を**表 5** に示す．

7.1 節のキー抽出の実験で，自動車ドメインについてはある程度の数の正解キーを獲得することができた．そこで，自動車ドメインについては，キー抽出実験で獲得したものを用いることにする．一方，ドリンクドメインについてはキー抽出実験で適切なキーを獲得することができなかった．しかし，適切なキーを取得できていれば，属性名のブート

表 5　各ドメインで用いた初期サイクルの正解属性名
Table 5　Attribute names used as initial seeds for each domain.

自動車ドメイン	ドリンクドメイン
Color	Taurine
Make	Caffeine
Displacement	Calcium
Mileage	Vitamin C
Height	Iron
Seating capacity	Sodium
Length	Carbonhydrate
Engine	Calories
Torque	Sugars
Curb weight	Protein

ストラッピングを行うことは可能である．そこで，ドリンクドメインについては，人手で適切なキーを与えることにした．

また，6 章の PC ドメインの実験において，属性名抽出の初期サイクルでキー抽出で獲得した正解キーをすべて用いる場合の方が，キー抽出のサイクルと属性名抽出のサイクルを同期させて動作させるよりも，より良い結果を得た．そこで，自動車ドメインではキー抽出で得た正解キーをすべて初期サイクルで用いることにし，ドリンクドメインでは人手で列挙した正解キーをすべて初期サイクルで用いることにした．

ドリンクドメインにおけるキーの与え方であるが，Amazon.com (usa) から，「Grocery & Gourmet Food」—「Beverrages」内の「Energy Drinks」と「Sports Drinks」のドリンク名を収集し，ランダムで選択している．PC ドメインで得た正解キーの数は 208 個，自動車ドメインで得た正解キーの数は 109 個であったので，ドリンクドメインについてもキーの個数を合わせるため，208 個用いた場合の結果と，109 個用いた場合の結果を示す．

7.2.2　実験結果

初期サイクルで，キー抽出で獲得したキーをすべて用いた場合，初期サイクルで 5 万ページに到達してしまった．属性名抽出手法では，獲得した属性名の確からしさを求めるため，確信度を算出している．そこで，確信度の上位個数ごとに精度を算出した．

図 15 に，結果を示す．すべてのドメインにおいて高い精度で属性名を抽出できていることが分かる．ドリンクドメインにおいて途中から精度が低下しているのは，いくつかのページにおいて，ドリンクドメインでない製品やサービスの名前を抽出してしまったからである．たとえば，ドリンクとその他食品は，非常に関連があるが，抽出された属性名には，“pasta”，“muffin”，“soy product”，“pan cake” などの食品名が多く含まれていた．また，ドリンクは他の医療サービスや製品との関連もあり，“anti-estrogen”（ホルモン系薬品）や “sleep aide”，“bone health” などの

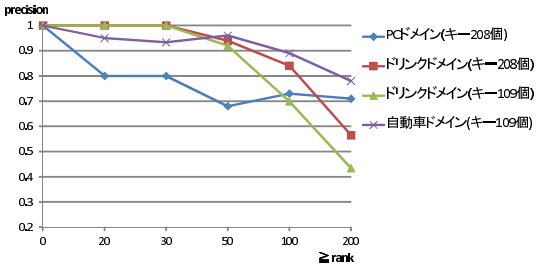


図 15　確信度の上位個数ごとの精度の推移
Fig. 15　Transition of precision according to the confidence rank for each domain.

表 6　キーを各サイクル時に与えた場合の処理時間（PC ドメイン）
Table 6　Processing time when inputting keys in each cycle (PC domain).

	処理時間（調査したパラメータの平均時間）
提案手法	18 分 46 秒
構造参照なし	17 分 55 秒
キー固定	17 分 11 秒
キーなし	19 分 32 秒

医療関係の製品やサービスの名称も多く含まれていた．自動車や PC と比べ，ドリンクはドメインとしての独立性が弱かったのが原因ではないかと考えられる．

なお，提案手法による属性名抽出のプロセスにおいて，抽出に要した時間は，**表 6**（キーを各サイクル時に与えた場合），**表 7**（すべてのキーを初期サイクル時に与えた場合）のとおりである．

7.2.3　従来手法との抽出精度の違い

最後に関連研究で示した従来手法と提案手法の，抽出精度の違いについて考察する．ここでは，関連研究の論文中で示された抽出精度と，本実験で示された提案手法の抽出精度を，実験条件の違いも考慮しつつ比較することとする．

関連研究の章で示したように，属性名抽出に関連する研究には以下の手法があげられる．

(1) 特定の表記パターンを用いて，属性名を抽出する手法
(2) 表やフォームの構造を利用して，属性名を抽出する手法
(3) Web 文書から表構造を特定し，属性名を抽出する手法
(4) クエリログを利用して，属性名を抽出する手法

ここでは，上記のタイプからそれぞれ代表的な研究事例を 1 つ取り上げ比較を行う．それぞれ手法の概要は 2 章で示したとおりであるので，ここでは割愛する．

(1) の代表例として，Tokunaga らの研究がある [23]．評価実験は，都市，博物館，川，自動車などのクラスに対して行っている．抽出した属性名が正解かどうかの判定は人手で行っている．その結果，抽出数が 50 個に対して，精度は 7 割程度となっている．我々の手法では，抽出数が 50 個に対して，精度は 9 割強の結果（自動車ドメインとドリンクドメイン）となっており，我々の手法の方が優れてい

表 7　キーを初期サイクルに与えた場合の処理時間
Table 7　Processing time when inputting keys at the initial cycle (PC domain).

	処理時間（調査したパラメータの時間）
PC ドメイン（キー 208 個）	1 時間 52 分 24 秒
ドリンクドメイン（キー 208 個）	1 時間 55 分 05 秒
ドリンクドメイン（キー 109 個）	1 時間 03 分 33 秒
自動車ドメイン（キー 109 個）	2 時間 50 分 47 秒

ると思われる．

(2) の代表例として，Tengli らの研究がある [14]．評価実験は，大学の Web ページ中の表を対象に，表の要素が属性名を表すものか，属性値を表すものか，それ以外かを判定している．評価結果は，属性名と属性値の判定結果を合わせたものを提示している．その結果，9 割程度の精度でその要素の分類が可能であることを示している．しかし，実験当時の大学の Web ページは比較的単純な構造の表が多く，手法の汎用性が示されているとは言い難い．また，表中には属性値に比べ属性名は少ないことと，属性値には数値が多く属性値を持つセルの判定は容易であると考えられるため，属性名のセルの判定精度はここまで高くないと思われる．

(3) の代表例として，Yin らの研究をあげる [19]．表が特定の実体に関連する属性–属性値を含んでいるかで評価した精度は 89.2%となっている．彼らは，表が特定できれば，その第 1 列（または第 1 行）を抽出すれば，属性名が抽出できると述べている．本研究でブートストラッピングを行う対象とすれば，より属性名の抽出精度が上がる可能性がある．

(4) の代表例として，Reisinger らの研究がある [24]．評価実験は，様々なドメインに適用しているが，本研究と共通するドメインである自動車で比較すると，彼らの手法では抽出数 50 個に対して，精度は 9 割強となっており，我々の手法とほぼ同程度の精度となっている．しかし，彼らの属性名の正解は，必ずしも明確な属性値が存在しないものも含まれる．たとえば，Religion（宗教）分野の message や TerroristGroup（テロリスト）分野の meaning などである．一方，我々の研究は抽出した属性名について，それに対応する明確な値が存在するかどうかを，Web で調べて判定している．そのため，我々の方が正解の判定基準が厳しいものとなっている．上記のように正解かどうかの判定基準が異なるため，どちらの手法が優れているかを述べることはできないが，彼らの手法とは異なる手法で，同程度の精度が得られたことには意味があると考えている．

最後に我々の手法は従来手法と異なる方法論をとっているため，これらを組み合わせて利用することも可能である．組み合わせることで，さらに精度や抽出数を向上させる可能性がある．

8. まとめと今後の予定

本研究ではブートストラッピングを用いて属性名を抽出する手法を提案した．本研究の技術的特徴は 3 つある．1 つ目は，2 つの属性名のそれぞれの前後にあるテキストで完全一致する部分を属性名抽出用のテンプレートとすることである．2 つ目は，上記テンプレートはそのテンプレートを作成したページにのみ適用し，さらに，文書構造を参照することで，テンプレートの適用範囲を限定することである．3 つ目は，属性名を獲得するブートストラッピングに加えて，属性名抽出の手がかりとなるキーを獲得するブートストラッピングも実行することである．

本論文ではまずキー抽出に着目し，どのキー抽出パラメータの組の場合に精度・抽出数が最良となるかを調査し，適切なキー抽出パラメータを得た．また，属性名抽出実験では，提案手法内で 4 つの手法を比較し，正解抽出数は構造参照なし手法が最良であった．提案手法において正解キーを初期サイクルに与える手法を試みた結果，正解属性名が 100 個以上増加した．最後に，ドリンクドメイン，自動車ドメインについても同様に実験を行い，提案手法の有効性を示した．属性名のブートストラッピングはこれらのドメインにおいても正しい属性名を抽出することができた．キーのブートストラッピングはドリンクドメインでは正しくキーを抽出することができなかった．ドメインによっては人手で正解キーを与える必要があることが分かった．今後の課題として属性名抽出を行う文書構造部分を文書全体を見て決定する方法の開発や，既存の表構造の判定方法と組み合わせる方法の開発があげられる．

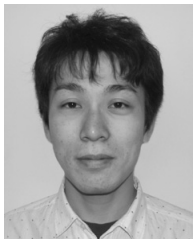
参考文献

[1] Codd, E.F.: *The relational model for database management: version 2*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA (1990).
[2] Appelt, D., Hobbs, J. and Israel, D.: Fastus: A finite-state processor for information extraction from real-world text, *Proc. IJCAI-93*, pp.1172–1178 (1993).
[3] Baumgartner, R., Flesca, S. and Gottlob, G.: Visual web information extraction with lixto, *Proc. 27th Very Large Databases Conference*, pp.119–128 (2001).
[4] Kushmerick, N., Weld, D. and Doorenbos, R.: Wrapper induction for information extraction, *Proc. 15th International Joint Conference on Artificial Intelligence*, pp.729–737 (1997).

[5] Ambite, J., Ashish, N., Barish, G., Knoblock, C., Minton, S., Modi, P., Muslea, I., Philpot, A. and Tejada, S.: Ariadne: A system for constructing mediators for internet sources, *Proc. ACM SIGMOD International Conference on Management of Data*, pp.561–563 (1998).
[6] 山田寛康，工藤　拓，松本裕治：Support vector machine を用いた日本語固有表現抽出, 情報処理学会論文誌, Vol.43, No.1, pp.43–53 (2002).
[7] Brin, S.: Extracting patterns and relations from the world wide web, *Proc. SIGMOD Workshop on Databases and the Web*, pp.172–183 (1998).
[8] Ciravegna, F., Chapman, S. and Dingli, A.: Learning to Harvest Information for the Semantic Web, *Proc. 1st European Semantic Web Symposium* (2004).
[9] Riloff, E. and Jones, R.: Learning dictionaries for information extraction by multi-level boot strapping, *Proc. 16th National Conference on Artificial Intellligence*, pp.474–479 (1999).
[10] Agichtein, E. and Gravano, L.: Snowball: Extracting relations from large plain-text collections, *Proc. 5th ACM International Conference on Digital Libraries*, pp.85–94 (2000).
[11] Raghavan, S. and Garcia-Molina, H.: Crawling the hidden web, *Proc. 27th Very Large Databases Conference*, pp.129–138 (2001).
[12] He, H., Meng, W., Yu, C. and Wu, Z.: Automatic integration of web search interfaces with WISE-Integrator, *VLDB Journal*, Vol.13, No.3, pp.256–273 (2004).
[13] Zhang, Z. and He, B.: Understanding web query interfaces: Best-Effort parsing with hidden syntax, *Proc. 2004 ACM SIGMOD international conference on Management of data*, pp.107–118 (2004).
[14] Tengli, A., Yang, Y. and Ma, N.: Learning table extraction from examples, *Proc. 2004 ACM SIGMOD International Conference on Management of Data*, pp.987–993 (2004).
[15] Chen, H. and Tsai, S.: Mining tables from large scale html texts, *Proc. 18th International Conference on Computational Linguistics*, pp.166–172 (2000).
[16] Cafarella, M.J., Halevy, A., Wang, D.Z., Wu, E. and Zhang, Y.: *Uncovering the relational web*, *WebDB'08* (2008).
[17] Cafarella, M.J., Halevy, A., Wang, D.Z., Wu, E. and Zhang, Y.: WebTables: Exploring the power of tables on the Web, *VLDB'08* (2008).
[18] Wang, Y. and Hu, J.: A machine learning based approach for table detection on the web, *WWW 2002* (2002).
[19] Yin, X., Tan, W. and Liu, C.: FACTO: A Fact Lookup Engine Based on Web Tables, *WWW 2011*, March 28, April 1, 2011, Hyderabad, India (2011).
[20] Yangarber, R. and Grishman, L.R.: Unsupervised learning of generalized names, *Proc. 19th International Conference on Computational Linguistics*, Vol.1, pp.474–479 (2002).
[21] 楠村幸貴，土方嘉徳，西田正吾：テンプレートの交差と DOM 構造の解析による情報抽出手法の提案，電子情報通信学会論文誌，Vol.J90-D, No.9, pp.2495–2509 (2007).
[22] The World Wide Web Consortium (W3C), available from ⟨http://www.w3.org/DOM/⟩.
[23] Tokunaga, K., Kazama, J. and Torisawa, K.: Automatic discovery of attribute words from web documents, *Proc. International Joint Conference of Natural Language Processing*, pp.106–118 (2005).
[24] Reisinger, J. and Pasca, M.: Low-Cost Supervision for Multiple-Source Attribute Extraction, *CICLing 2009*, LNCS 5449, pp.382–393 (2009).
[25] 吉永直樹，鳥澤健太郎：Web からの属性情報記述ページの発見，人工知能学会論文誌，Vol.21, No.6, pp.493–501 (2006).

野村 慎太郎

2011 年大阪大学基礎工学部システム科学科卒業．2013 年同大学大学院修士課程修了．2013 年シャープ株式会社入社．2012 年 WebDB フォーラム 2011 優秀論文賞，学生奨励賞，各受賞．在学中は，情報抽出の研究に従事．



中根 史敬

2007 年大阪大学基礎工学部システム科学科卒業．2009 年同大学大学院修士課程修了．2009 年パナソニック株式会社入社．2012 年 WebDB フォーラム 2011 優秀論文賞受賞．在学中は，情報抽出の研究に従事．

土方 嘉徳　（正会員）

1996 年大阪大学基礎工学部システム工学科卒業．1998 年同大学大学院修士課程修了．同年日本アイ・ビー・エム（株）東京基礎研究所入社．2002 年より大阪大学大学院基礎工学研究科システム創成専攻助手．2009 年より同准教授．2005 年インタラクション 2005 ベストペーパー賞，2006 年 ACM IUI Best Paper Award，DEWS2006 優秀論文賞，2011 年 WebDB フォーラム 2011 最優秀論文賞，2012 年 WebDB フォーラム 2012 優秀論文賞，2013 年インタラクション 2013 ベストペーパー賞，情報処理学会山下記念研究賞，各受賞．情報推薦，Web インテリジェンス，テキストマイニングの研究に従事．人工知能学会，ヒューマンインタフェース学会，日本データベース学会ほか会員．電子情報通信学会シニア会員．博士（工学）．

西田 正吾　（正会員）

1952 年 1 月 5 日生．1974 年 3 月東京大学工学部電子工学科卒業．1976 年 3 月同大学大学院修士課程修了．同年 4 月三菱電機（株）入社．同社中央研究所システム基礎研究部研究員，グループマネージャーを経て，1995 年 4 月大阪大学基礎工学部教授．その後，大阪大学大学院基礎工学研究科長・基礎工学部長，理事・副学長を経て，現在，大阪大学大学院基礎工研究科教授．システム技術，ヒューマンインタフェース技術，メディア技術の研究に従事．1984〜1985 年 MIT メディアラボ客員研究員．ヒューマンインタフェース学会論文賞（2001 年，2005 年），電気学会業績賞（2004 年），船井情報科学振興賞（2006 年）等受賞．IEEE Fellow．電子情報通信学会フェロー．電気学会フェロー．著書は，『ヒューマン・コンピュータ交流技術』（オーム社，共著），『メディア工学』（朝倉書店），『情報メディア工学』（オーム社，共著）等．工学博士．

（担当編集委員　藤井 敦）